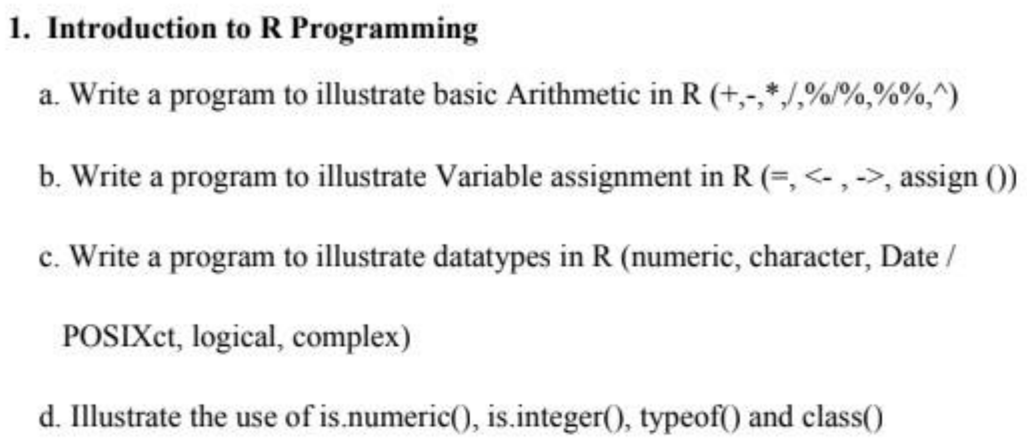
ASSIGNMENT-1

NAME: kK.RAGHU
REG NO:22MIS7195



1. Introduction to R Programming

a. Write a program to illustrate basic Arithmetic in R (+,-,*,/,%/%,%%,^)

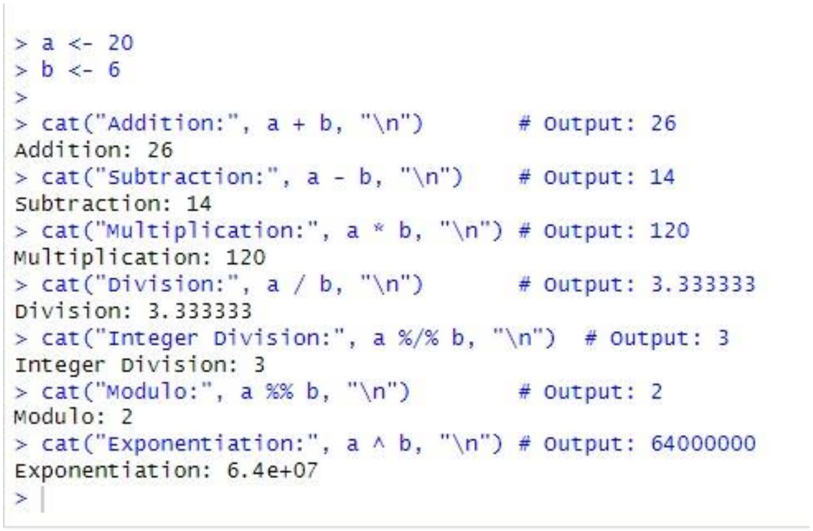
b. Write a program to illustrate Variable assignment in R (=, <- , ->, assign ())

c. Write a program to illustrate datatypes in R (numeric, character, Date /

POSIXct, logical, complex)

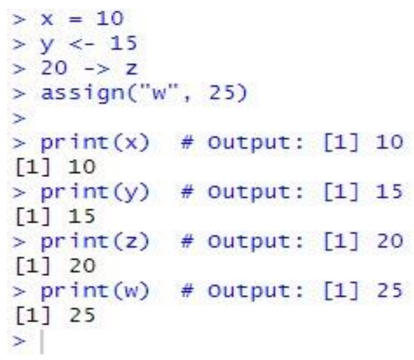
d. Illustrate the use of is.numeric(), is.integer(), typeof() and class()

OUTPUT:
a.



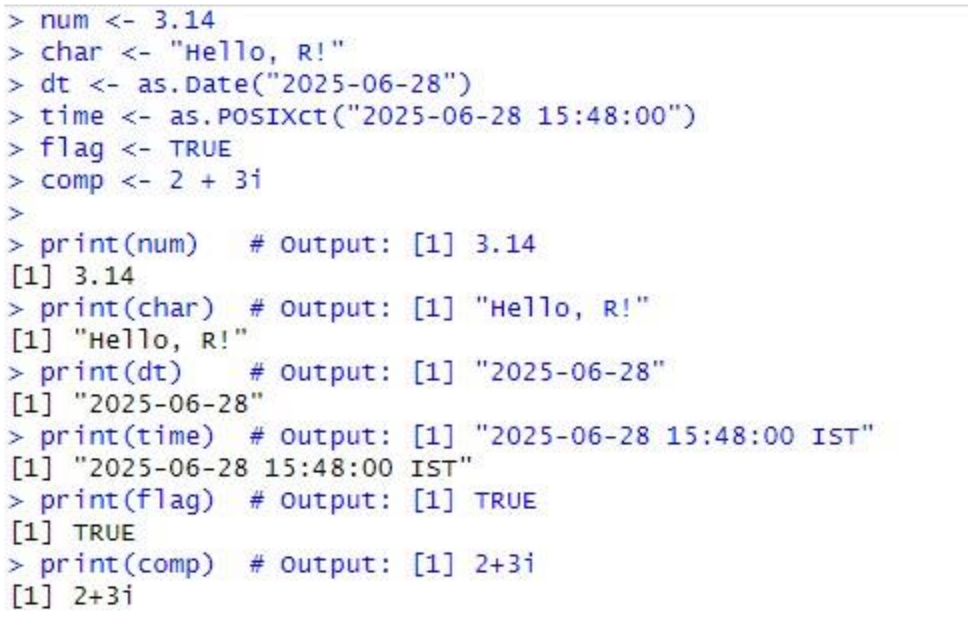
```
> a <- 20
> b <- 6
>
> cat("Addition:", a + b, "\n")       # Output: 26
Addition: 26
> cat("Subtraction:", a - b, "\n")    # Output: 14
Subtraction: 14
> cat("Multiplication:", a * b, "\n") # Output: 120
Multiplication: 120
> cat("Division:", a / b, "\n")       # Output: 3.333333
Division: 3.333333
> cat("Integer Division:", a %/% b, "\n")  # Output: 3
Integer Division: 3
> cat("Modulo:", a %% b, "\n")        # Output: 2
Modulo: 2
> cat("Exponentiation:", a ^ b, "\n") # Output: 64000000
Exponentiation: 6.4e+07
> |
```

b.



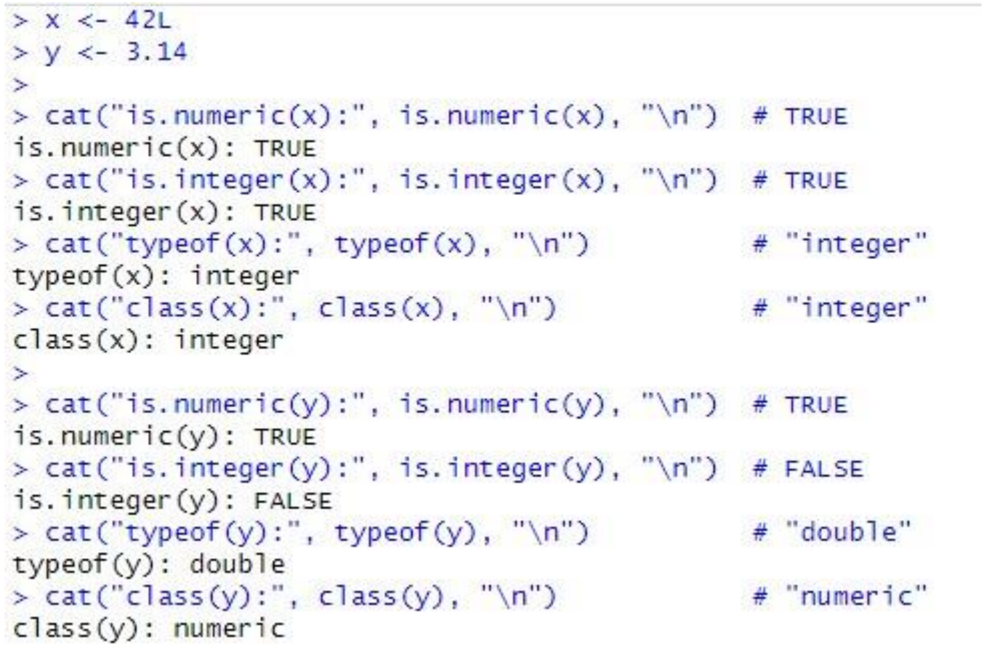
```
> x = 10
> y <- 15
> 20 -> z
> assign("w", 25)
>
> print(x)  # Output: [1] 10
[1] 10
> print(y)  # Output: [1] 15
[1] 15
> print(z)  # Output: [1] 20
[1] 20
> print(w)  # Output: [1] 25
[1] 25
> |
```

c.



```
> num <- 3.14
> char <- "Hello, R!"
> dt <- as.Date("2025-06-28")
> time <- as.POSIXct("2025-06-28 15:48:00")
> flag <- TRUE
> comp <- 2 + 3i
>
> print(num)   # Output: [1] 3.14
[1] 3.14
> print(char)  # Output: [1] "Hello, R!"
[1] "Hello, R!"
> print(dt)    # Output: [1] "2025-06-28"
[1] "2025-06-28"
> print(time)  # Output: [1] "2025-06-28 15:48:00 IST"
[1] "2025-06-28 15:48:00 IST"
> print(flag)  # Output: [1] TRUE
[1] TRUE
> print(comp)  # Output: [1] 2+3i
[1] 2+3i
```

d.



```
> x <- 42L
> y <- 3.14
>
> cat("is.numeric(x):", is.numeric(x), "\n")  # TRUE
is.numeric(x): TRUE
> cat("is.integer(x):", is.integer(x), "\n")  # TRUE
is.integer(x): TRUE
> cat("typeof(x):", typeof(x), "\n")          # "integer"
typeof(x): integer
> cat("class(x):", class(x), "\n")            # "integer"
class(x): integer
>
> cat("is.numeric(y):", is.numeric(y), "\n")  # TRUE
is.numeric(y): TRUE
> cat("is.integer(y):", is.integer(y), "\n")  # FALSE
is.integer(y): FALSE
> cat("typeof(y):", typeof(y), "\n")          # "double"
typeof(y): double
> cat("class(y):", class(y), "\n")            # "numeric"
class(y): numeric
```

2. Getting Used to R: working with more datatypes

a. Vectors and its operations

Create a vector X of numbers 23, 54, 78, 9, 10, 36, 72, 11, 19, 4

i) Create new vector Y which holds all X>=25

ii) Create new vector Z which holds all X divisible by 6

iii) Create new vector M which holds all X except 1st, 3rd and 6th elements

iv) Create new vector N which holds 2nd, 4th and 6th to 8th elements of X

b. Write a R program to create a vector of a specified type and length. Create vector of numeric, complex, logical and character types of length 6.

```
Ex: x = vector("numeric", 5)
print("Numeric Type:")
print(x)
```

c. Write a R program to add two vectors of integers type of length 3. Print original and resultant vector.

d. Write a R program to append value to a given vector.

```
x = c(10, 20, 30)
x=c(x,34)
print(x)
```

e. Write a R program to multiply two vectors of integers type and length 3.

f. Write a R program to divide two vectors of integers type and length 3.

g. Write a R program to find Sum, Mean and Product of a Vector.

h. Write a R program to find Sum, Mean and Product of a Vector, ignore element like NA or NULL

i. Write a R program to find the minimum and the maximum of a Vector. (min(),max())

j. Write a R program to sort a Vector in ascending and descending order.

k. Write a R program to test whether a given vector contains a specified element.

```
Ex: x = c(10, 20, 30, 25, 9, 26)
is.element(25, x)
TRUE
```

l. Write a R program to count the specific value in a given vector.

m. Write a R program to find common elements from multiple vectors.

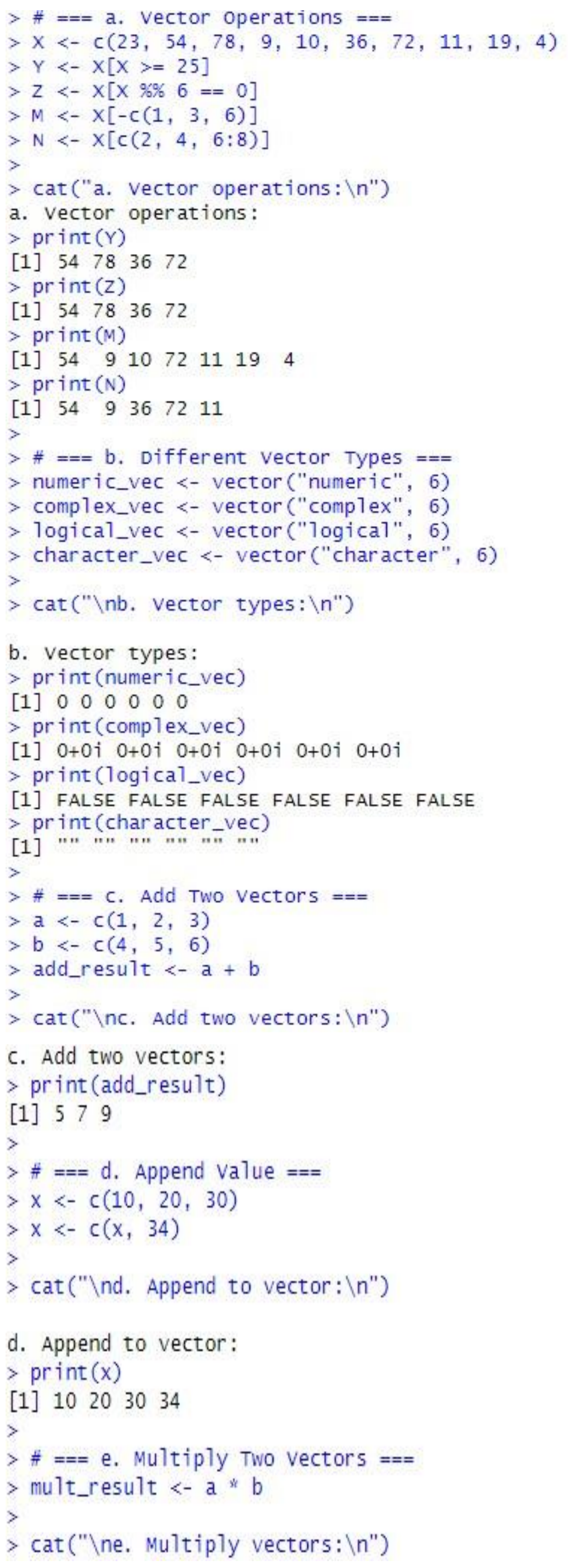
intersect(x,y), where x,y are vectors

n. Write a R program to list the distinct values in a vector from a given vector.

print(unique(v)), where v is a vector

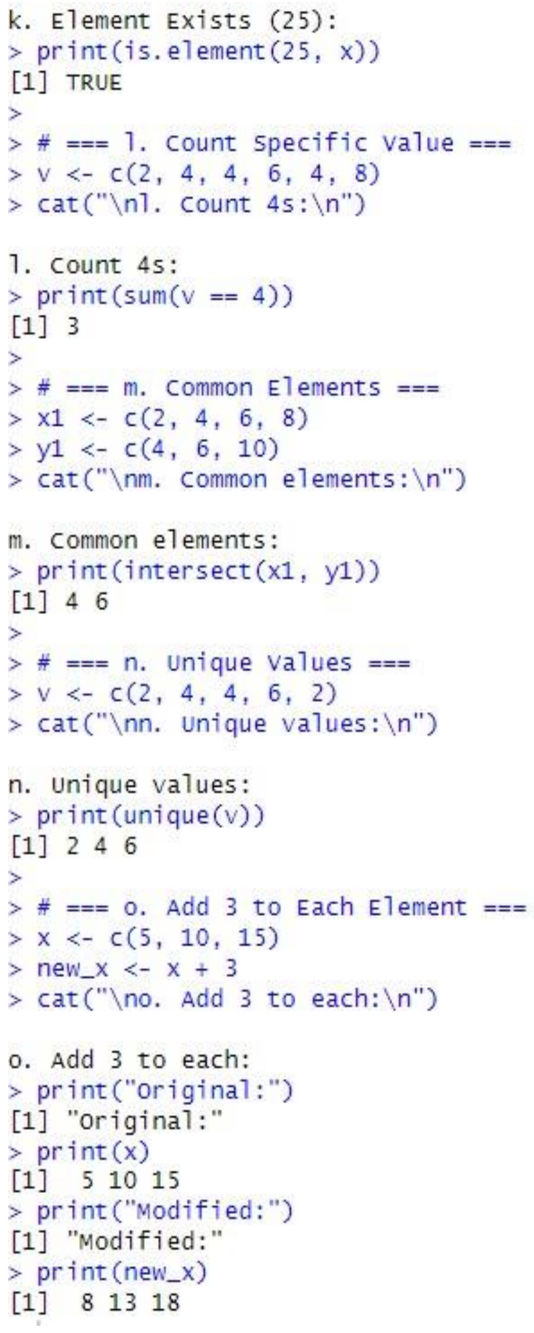
o. Write a R program to add 3 to each element in a given vector and store the result in the w vector.Print the original and new vector.

OUTPUT:



```
> # === a. Vector Operations ===
> X <- c(23, 54, 78, 9, 10, 36, 72, 11, 19, 4)
> Y <- X[X >= 25]
> Z <- X[X %% 6 == 0]
> M <- X[-c(1, 3, 6)]
> N <- X[c(2, 4, 6:8)]
> 
> cat("a. Vector operations:\n")
a. Vector operations:
> print(Y)
[1] 54 78 36 72
> print(Z)
[1] 54 78 36 72
> print(M)
[1] 54  9 10 72 11 19  4
> print(N)
[1] 54  9 36 72 11
> 
> # === b. Different Vector Types ===
> numeric_vec <- vector("numeric", 6)
> complex_vec <- vector("complex", 6)
> logical_vec <- vector("logical", 6)
> character_vec <- vector("character", 6)
> 
> cat("\nb. Vector types:\n")

b. Vector types:
> print(numeric_vec)
[1] 0 0 0 0 0 0
> print(complex_vec)
[1] 0+0i 0+0i 0+0i 0+0i 0+0i 0+0i
> print(logical_vec)
[1] FALSE FALSE FALSE FALSE FALSE FALSE
> print(character_vec)
[1] "" "" "" "" "" ""
> 
> # === c. Add Two Vectors ===
> a <- c(1, 2, 3)
> b <- c(4, 5, 6)
> add_result <- a + b
> 
> cat("\nc. Add two vectors:\n")

c. Add two vectors:
> print(add_result)
[1] 5 7 9
> 
> # === d. Append Value ===
> x <- c(10, 20, 30)
> x <- c(x, 34)
> 
> cat("\nd. Append to vector:\n")

d. Append to vector:
> print(x)
[1] 10 20 30 34
> 
> # === e. Multiply Two Vectors ===
> mult_result <- a * b
> 
> cat("\ne. Multiply vectors:\n")
```

```
e. Multiply vectors:
> print(mult_result)
[1]  4 10 18
>
> # === f. Divide Two Vectors ===
> div_result <- a / b
>
> cat("\nf. Divide vectors:\n")

f. Divide vectors:
> print(div_result)
[1] 0.25 0.40 0.50
>
> # === g. Sum, Mean, Product ===
> v <- c(2, 4, 6)
> cat("\ng. Sum, Mean, Product:\n")

g. Sum, Mean, Product:
> print(sum(v))
[1] 12
> print(mean(v))
[1] 4
> print(prod(v))
[1] 48
>
> # === h. Handle NA and NULL ===
> v <- c(2, NA, 4, 6)
> cat("\nh. With NA:\n")

h. With NA:
> print(sum(v, na.rm = TRUE))
[1] 12
> print(mean(v, na.rm = TRUE))
[1] 4
> print(prod(v, na.rm = TRUE))
[1] 48
>
> # === i. Min and Max ===
> v <- c(10, 25, 3, 70)
> cat("\ni. Min and Max:\n")

i. Min and Max:
> print(min(v))
[1] 3
> print(max(v))
[1] 70
>
> # === j. Sort Vector ===
> v <- c(9, 3, 12, 6)
> cat("\nj. Sort:\n")

j. Sort:
> print(sort(v))                # Ascending
[1]  3  6  9 12
> print(sort(v, decreasing=TRUE)) # Descending
[1] 12  9  6  3
>
> # === k. Check for Element ===
> x <- c(10, 20, 30, 25, 9, 26)
> cat("\nk. Element Exists (25):\n")
```

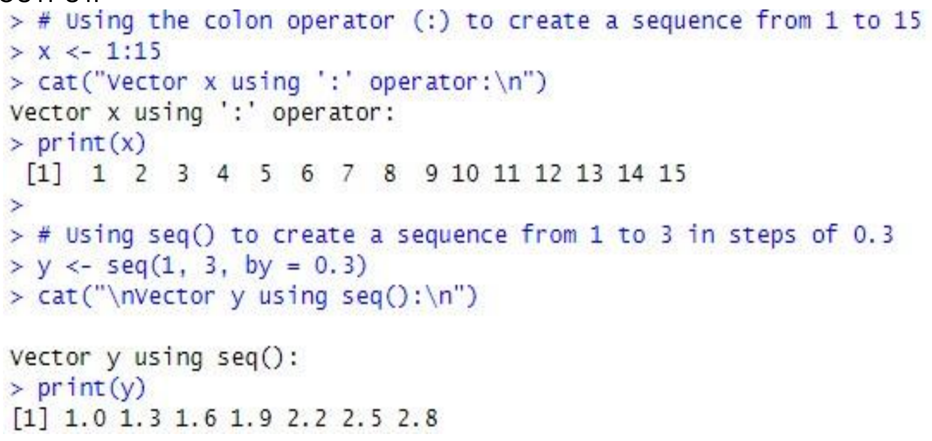


```
k. Element Exists (25):
> print(is.element(25, x))
[1] TRUE
>
> # === l. Count Specific Value ===
> v <- c(2, 4, 4, 6, 4, 8)
> cat("\nl. Count 4s:\n")

l. Count 4s:
> print(sum(v == 4))
[1] 3
>
> # === m. Common Elements ===
> x1 <- c(2, 4, 6, 8)
> y1 <- c(4, 6, 10)
> cat("\nm. Common elements:\n")

m. Common elements:
> print(intersect(x1, y1))
[1] 4 6
>
> # === n. Unique Values ===
> v <- c(2, 4, 4, 6, 2)
> cat("\nn. Unique values:\n")

n. Unique values:
> print(unique(v))
[1] 2 4 6
>
> # === o. Add 3 to Each Element ===
> x <- c(5, 10, 15)
> new_x <- x + 3
> cat("\no. Add 3 to each:\n")

o. Add 3 to each:
> print("Original:")
[1] "Original:"
> print(x)
[1]  5 10 15
> print("Modified:")
[1] "Modified:"
> print(new_x)
[1]  8 13 18
```

17. Write a R program to create a vector using : operator and seq() function.

```
y = seq(1, 3, by=0.3)

x = 1:15
```

OUTPUT:



```
> # Using the colon operator (:) to create a sequence from 1 to 15
> x <- 1:15
> cat("Vector x using ':' operator:\n")
Vector x using ':' operator:
> print(x)
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
>
> # Using seq() to create a sequence from 1 to 3 in steps of 0.3
> y <- seq(1, 3, by = 0.3)
> cat("\nVector y using seq():\n")

Vector y using seq():
> print(y)
[1] 1.0 1.3 1.6 1.9 2.2 2.5 2.8
```